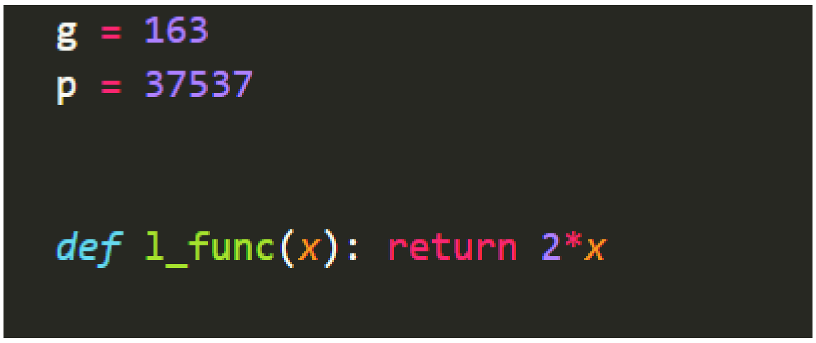
Task 1 PRG

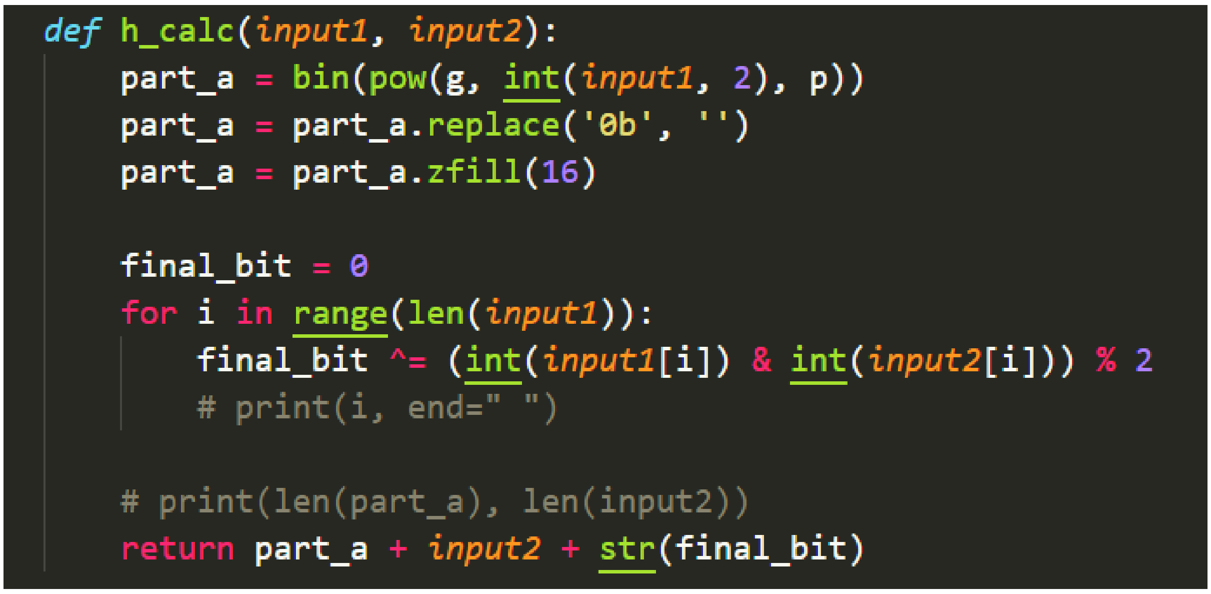


```python
g = 163
p = 37537


def l_func(x): return 2*x
```

L_func()

Function takes in the size of the input seed as a parameter, and outputs monotonically larger length, which is used as how many times the loop should iterate over and over and ultimately decides the size of the output.



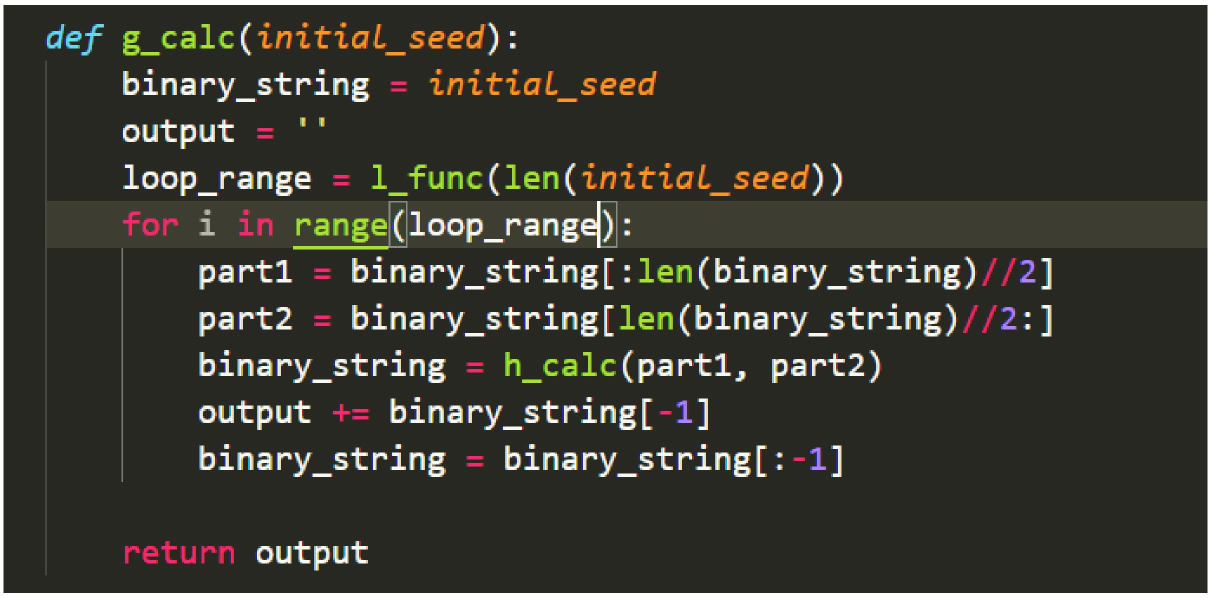
```python
def h_calc(input1, input2):
    part_a = bin(pow(g, int(input1, 2), p))
    part_a = part_a.replace('0b', '')
    part_a = part_a.zfill(16)

    final_bit = 0
    for i in range(len(input1)):
        final_bit ^= (int(input1[i]) & int(input2[i])) % 2
        # print(i, end=" ")

    # print(len(part_a), len(input2))
    return part_a + input2 + str(final_bit)
```

H_calc()

Function takes as input the 2 parts of seed provided as input from the user. It is responsible for taking the n bits as input and outputs n+1 bits to the calling function, with the n+1th bit being the hardcore_predicate.



```python
def g_calc(initial_seed):
    binary_string = initial_seed
    output = ''
    loop_range = l_func(len(initial_seed))
    for i in range(loop_range):
        part1 = binary_string[:len(binary_string)//2]
        part2 = binary_string[len(binary_string)//2:]
        binary_string = h_calc(part1, part2)
        output += binary_string[-1]
        binary_string = binary_string[:-1]

    return output
```

G_calc()

Takes initial_seed as input from the user, it acts as the driver function of the code. The final output returned is a bit string of length determined by the l_func() and is a random number that can't be discerened by a probabilistic TM, on whether it's pseudorandom or random.

Sample Input

Seed: 11101011

Sample Output

0001101001101011